SQL Server e MySQL

SQL Server

Viste

Le viste in SQL Server sono oggetti che permettono di visualizzare una porzione di una o più tabelle, sotto forma di una tabella virtuale. Le viste non contengono dati, ma rappresentano una vista personalizzata dei dati contenuti nelle tabelle sottostanti. In pratica, le viste sono come una finestra su un insieme di dati di una o più tabelle, e consentono di manipolare questi dati senza dover modificare le tabelle stesse.

Per **creare una vista in SQL Server**, è possibile utilizzare il comando `CREATE VIEW`, seguito dal nome della vista e dalla definizione della vista stessa. Ad esempio, per creare una vista che mostri solo i nomi dei clienti di una tabella "Clienti", è possibile utilizzare il seguente codice SQL:

```
CREATE VIEW VistaClienti AS
SELECT NomeCliente FROM Clienti;
```

Una volta creata la vista, è possibile utilizzarla come se fosse una tabella normale. Ad esempio, per visualizzare i dati della vista appena creata, è possibile utilizzare il

comando `SELECT`:

```
SELECT * FROM VistaClienti;
```

Configurazione di un database

La configurazione di un database in SQL Server dipende dalle esigenze specifiche dell'applicazione e delle funzionalità richieste. In generale, è possibile configurare il database utilizzando lo strumento di gestione SQL Server Management Studio (SSMS), che consente di configurare vari aspetti del database, come ad esempio le opzioni di sicurezza, le impostazioni di backup, le proprietà del database e così via.

Per creare tabelle in SQL Server, è possibile utilizzare il comando `CREATE TABLE`, seguito dal nome della tabella e dalle definizioni delle colonne. Ad esempio, per creare una tabella "Clienti" con le colonne "IDCliente", "NomeCliente" e "CognomeCliente", è possibile utilizzare il seguente codice SQL:

```
CREATE TABLE Clienti (
    IDCliente INT PRIMARY KEY,
    NomeCliente VARCHAR(50),
    CognomeCliente VARCHAR(50)
);
```

In questo esempio, la colonna "IDCliente" è definita come chiave primaria della tabella. Una volta creata la tabella, è possibile inserire i dati utilizzando il comando `INSERT`, oppure modificare i dati utilizzando il comando `UPDATE`.

MySQL

Creazione viste:
In SQL Server, la creazione di una vista si basa sulla selezione di una porzione di una o più tabelle, e la sua definizione viene salvata come oggetto separato nel database. Per creare una vista, è possibile utilizzare il comando `CREATE VIEW`, specificando il nome della vista e la definizione della selezione. Ad esempio, per creare una vista che mostra solo i nomi e i cognomi dei clienti nella tabella "Clienti", è possibile utilizzare il seguente comando SQL:

```
CREATE VIEW VistaClienti AS
SELECT NomeCliente, CognomeCliente
FROM Clienti;
```

Definizione dei dati

In SQL Server, la definizione dei dati viene effettuata attraverso l'utilizzo del comando `CREATE TABLE`, che permette di creare una nuova tabella all'interno del database. La definizione della tabella prevede la specifica dei nomi delle colonne e dei tipi di dati associati ad ognuna di esse. Ad esempio, per creare una tabella "Ordini" con le colonne "IDOrdine", "DataOrdine" e "ImportoOrdine", si può utilizzare il seguente comando SQL:

```
CREATE TABLE Ordini (
    IDOrdine INT PRIMARY KEY,
    DataOrdine DATE,
    ImportoOrdine DECIMAL(10,2)
);
```

GROUP BY

Il comando `GROUP BY` viene utilizzato per raggruppare i record in base al valore di una o più colonne della tabella, restituendo un set di risultati aggregati. Ad esempio, per ottenere il totale degli ordini per ciascun cliente nella tabella "Ordini", è possibile utilizzare il seguente comando SQL:

```
SELECT IDCliente, SUM(ImportoOrdine) as TotaleOrdini
FROM Ordini
GROUP BY IDCliente;
```

HAVING

Il comando `HAVING` viene utilizzato per filtrare i risultati di una query che utilizza il comando `GROUP BY`. In pratica, consente di specificare una condizione di filtro che viene applicata ai risultati aggregati. Ad esempio, per ottenere solo i clienti che hanno effettuato ordini per un importo superiore a 1000, è possibile utilizzare il seguente comando SQL:

```
SELECT IDCliente, SUM(ImportoOrdine) as TotaleOrdini
FROM Ordini
GROUP BY IDCliente
HAVING SUM(ImportoOrdine) > 1000;
```

IN, EXISTS e ALL

IN, EXISTS e ALL sono operatori di comparazione che possono essere utilizzati in SQL per filtrare i dati in una query. Ecco una breve descrizione di ciascun operatore:

- **IN**: l'operatore `IN` viene utilizzato per confrontare un valore con un set di valori definiti in una sottoquery. In pratica, l'operatore restituisce i record che contengono un valore corrispondente a uno dei valori presenti nella sottoquery. Ad esempio, la seguente query restituirebbe tutti i record della tabella "clienti" che corrispondono a uno dei valori presenti nella sottoquery:

```
SELECT *
FROM clienti
WHERE id_cliente IN (1, 2, 3);
```

- **EXISTS**: l'operatore `EXISTS` viene utilizzato per verificare l'esistenza di almeno un record in una sottoquery. In pratica, l'operatore restituisce i record della tabella principale solo se esiste almeno un record corrispondente nella sottoquery. Ad esempio, la seguente query restituirebbe tutti i record della tabella "ordini" che corrispondono a un cliente presente nella tabella "clienti":

```
SELECT *
FROM ordini o
WHERE EXISTS (
    SELECT 1
    FROM clienti c
    WHERE c.id_cliente = o.id_cliente
);
```

- **ALL**: l'operatore `ALL` viene utilizzato per confrontare un valore con tutti i valori presenti in una sottoquery. In pratica, l'operatore restituisce i record che corrispondono al confronto con tutti i valori della sottoquery. Ad esempio, la

seguente query restituirebbe tutti i record della tabella "prodotti" che hanno un prezzo superiore a tutti i prodotti della categoria "alimentari":

```
SELECT *
FROM prodotti
WHERE prezzo > ALL (
    SELECT prezzo
    FROM prodotti
    WHERE categoria = 'alimentari'
);
```